1. Monte Carlo methods, and the Ising model

Monte-Carlo algorithms (named after the Monte Carlo casino) work by using random number to estimate integral and statistical averages. In this Section we will study how they work, and we will focus on the Metropolis algorithm which is the primary way to estimate a quantity in a system in thermodynamic equilibrium. The physical system which we consider as an example is the Ising model which you should study in Checkpoint 1, and which you should have encountered in Statistical Physics – however the basic properties are briefly reviewed here as well. The approach taken here is practical, and the aim is to allow you to understand (i) how to set up in practice a Monte Carlo simulation based on the Metropolis algorithm, and (ii) why it works. We will also briefly discuss how to estimate errors in the quantities you measure in a Monte-Carlo simulation.

1.1. Monte-Carlo integration

Let us suppose that we wish to evaluate numerically the following d-dimensional integral, over a volume V,

$$I = \int_V d^d\mathbf{r} f(\mathbf{r}) = \int dx_1 \dots \int dx_d \theta_V(\mathbf{r}) f(x_1, \dots x_d) \tag{1}$$

where f is some given function, and we have introduced a function $\theta_V(\mathbf{r})$ which is 1 inside V and 0 outside (sometimes this is called the indicator function for V).

One way to estimate I numerically is via the trapezoid rule. This consists in coming up with a subdivision of the volume V into n intervals, each one of volume δ^d, so that $n = V/\delta^d$. Inside each interval, we approximate f via a linear function (in all of its arguments), thereby making an error of order δ^2. We then evaluate I via the sum

$$I \simeq \sum_{i=1}^{n} \delta^d \overline{f_i} \tag{2}$$

where $\overline{f_i}$ is the average of f over the i-th integration volume. A little thought shows that the error on I should be of order $n\delta^d\delta^2$ as there are n terms and each of the $\overline{f_i}$ has an error δ^2 as it is a linearly accurate approximation to f. Recalling that $n = V/\delta^d$, we then get that the error is proportional to δ^2 or to $n^{-2/d}$. Apart from the algebra, the important thing to get is that the error made by estimating a d-dimensional integral with a standard discretisation into a sum (with the trapexoidal rule) decreases with the number of integration intervals n, as intuition suggests, but the speed of convergence depends on d, and gets slower as the dimension increases.

In other words, in high dimension standard discretisation is not a good way to compute integrals – it also becomes very slow! Monte-Carlo integration provides an alternative. The underlying idea is to relate I to a suitable average,

as follows,

$$I = \int_V d^d\mathbf{r} f(\mathbf{r}) = \frac{\int_V d^d\mathbf{r} f(\mathbf{r})}{V} V = \langle f \rangle V \tag{3}$$

where $\langle f \rangle$ is nothing but the average of the function $f(\mathbf{r})$ over the volume V. Then a possible way to estimate I is to draw a few (say n) random numbers, uniformly distributed within the d-dimensional volume V, and attempt an estimate of the average of f within the volume using these values. If we label by $\{\mathbf{r}_1, \ldots, \mathbf{r}_n\}$ the n points chosen randomly in V, the Monte-Carlo estimation of the integral would then be

$$I \simeq \frac{\sum_{i=1}^n f(\mathbf{r}_i)}{n} V. \tag{4}$$

In other words, it is a bit like doing an experiment, in which we measure f several times and then average out the measurements to find $\langle f \rangle$. It might seem that this is a rather inefficient way to estimate our integral! However, we can convince ourselves quite easily that the error in the average in Eq. 4 decreases with n as $n^{-1/2}$ (as the standard error of the mean has this dependence, due ultimately to the central limit theorem). Therefore, when the dimension is high (or indeed larger than 4), it appears Monte-Carlo methods can estimate integrals better than standard discretisation methods.

1.2. Importance sampling and the Ising model

An example of a situation in which we need to routinely perform integrals in (very) high dimensions is provided by statistical mechanics. In this field, Monte Carlo methods are very much used, and are quite powerful. Recall that the partition function of a system with N particles/constituents is given by

$$Z_N = \sum_{\mu \in \{\text{states}\}} \exp\left(-\beta E_\mu\right) \tag{5}$$

where E_μ is the energy of the system in state μ. [The energy clearly depends on which state (or configuration) is considered.] Now what the sum over states means depends on the specific system we are studying. In all cases, however, it corresponds to a very difficult operation to perform numerically. For a three-dimensional fluid of N interacting particles, the sum is an integral over all particle positions (and momenta, but the integral there can easily be performed analytically). Therefore one is left with a $3N$ dimensional integral, where ideally N is very large! Monte-Carlo methods are clearly very good to estimate integrals like these.

In particular, we are typically interested in the average of some observable quantity A (for instance the energy, or its square), so that we would like to compute, for example,

$$\langle A \rangle = \frac{\sum_{\mu \in \text{states}} A_\mu \exp\left(-\beta E_\mu\right)}{\sum_{\mu \in \text{states}} \exp\left(-\beta E_\mu\right)}, \tag{6}$$

where A_μ is the value of the quantity A in state μ. A Monte-Carlo strategy to estimate this integral is to generate randomly n states, uniformly distributed among the ensemble of all possible configuration of the system, and average both the numerator and denominator in Eq. 6, as follows

$$\langle A \rangle = \frac{\sum_{i=1}^{n} A_i \exp\left(-\beta E_i\right)}{\sum_{i=1}^{n} \exp\left(-\beta E_i\right)}, \tag{7}$$

with E_i and A_i the value of the energy and of the quantity A in the i-th generated state.

As a prototype statistical physics system, we will consider the Ising model. The Ising model comprises N spins S_i (now therefore $i = 1, \ldots, N$) on a lattice, each of which can point up, $S_i = 1$, or down, $S_i = -1$. Each neighbouring pair of aligned spins lowers the energy of the system by an amount $J > 0$. Thus, given a spin configuration $\{S_i\}$ (i.e., a state of the system), the total energy is

$$E(\{S_i\}) = -J \sum_{\langle ij \rangle} S_i S_j \tag{8}$$

where the sum is over all distinct nearest neighbour pairs $\langle ij \rangle$. According to the Boltzmann distribution, that probability of observing a given configuration $\{S_i\}$ at equilibrium is

$$P(\{S_i\}) = \exp\left[-E(\{S_i\})/(k_B T)\right] \tag{9}$$

where k_B is Boltzmanns constant and T is the temperature. In line with standard practice, we will call $\beta = \frac{1}{k_B T}$ in the following. Now, in Eqs. 5 and 6 the sum over states is a discrete sum, rather than a continuum integral, but the simplest Monte-Carlo way to evaluate the average of an observable A is still to generate n uncorrelated conformations of N spin randomly, in a way in which all configurations are equally likely, and then to compute $\langle A \rangle$ as in Eq. 7. A simple way to generate states randomly and uniformly is to draw a random number between 0 and 1 for each site of the N sites in the lattice, and assign the spin value $+1$ is the random number is larger or equal to $1/2$, and -1 otherwise.

There is however an important fundamental issue with any Monte-Carlo estimate of averages of observables via Eq. 7, which we now need to discuss. The problem is that, if we generate states randomly with uniform probability, so that each configuration is *a priori* equally likely, we are in danger of generating several configurations for which the associated Boltzmann weight, i.e. the value of $\exp\left[-E/(k_B T)\right]$, is small, or even extremely small with respect to the maximal value of $\exp\left[-E/(k_B T)\right]$ over the whole configuration space. Think for instance of an Ising model at low T. Here the dominating states are the two degenerate ground states, where all spins are aligned (all up or all down). However, there are as many as 2^N states from which to choose from: the chance of hitting by chance either of the two ground state (or a closely related state) for a large system is practically zero! Then, the estimate of $\langle A \rangle$ will never be good.

This problem can be solved by "importance sampling". This refers to the idea to generate configurations not randomly, but in a biased way, and according to a predefinite probability distribution. Imagine that, instead of generating configurations $\gamma_1, \ldots, \gamma_n$ with uniform probability, we generate $\mu_1, \ldots, \mu_n$ with probability corresponding to the Boltzmann distribution, i.e. with probability proportional to the exponential of (minus) the energy of the configuration divided by $k_B T$. In this way we will automatically sample more often the states which have a larger weight in the sums in both the numerator and denominator of Eq. 7, and our estimate of $\langle A \rangle$ should be much better. If the $\{\mu_i\}_i$ configurations are generated according to importance sampling (guided by the Boltzmann distribution), then Eq. 6 can be estimated as a simple average as follows,

$$\langle A \rangle = \frac{\sum_{i=1}^{n} A(\mu_i)}{n}. \tag{10}$$

This is the way that most Monte-Carlo simulations are set up, especially when formulated to solve problems in statistical physics (e.g. to study our Ising model).

1.3. Markov chains and the Metropolis algorithm

How can we possibly choose a set of states $\{\mu_i\}$ according to the Boltzmann (or for what matters, any other set equilibrium) distribution?

The idea is to set up a "Markov chain", i.e. a dynamical rule which, starting from a given state μ, prescribes a way to choose (stochastically) a following state ν, and from there another one, λ, and so on and so forth. Provided the dynamical rule is chosen "appropriately", then after several steps the states are generated with a steady state (equilibrium) probability which is the one we want.

Let us consider as an example the Ising model introduced previously, and describe one possible "appropriate" way to generate a Markov chain so as to sample states according to the Boltzmann weight. Imagine starting from a state μ given in Fig. 1A.

1. We first generate a *candidate* new state, ν, by choosing one spin at random, and flipping it. This is called "Glauber dynamics". The proposed state is shown in Fig. 1B.

2. We then compute the energy of both the original state, E_μ, and that of the proposed new state, with the spin flipped, E_ν. We note that, due to the local nature of the Glauber update, it is sensible to not compute energies by summing over all nearest neighbour links, but to only consider the variation of the energy associated with the links affected by the flip. As an exercise, try computing the change of energy $\Delta E = E_\nu - E_\mu$ by only summing over local quantities involving spins affected by the flip (see Checkpoint 1).

3. Now that we have a proposed new state ν, and an energy difference $\Delta E = E_\nu - E_\mu$, we need to provide a rule as to whether or not the new configuration is accepted. This is done according to the so-called Metropolis algorithm,

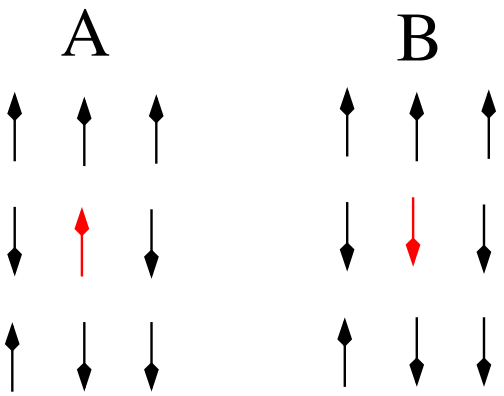

Figure 1: (A) A state of a 3×3 Ising lattice, say μ. (B) Another state ν obtained from μ via Glauber dynamics, selecting the red spin in (A,B) and flipping it.

which is arguably the most important concept in Monte-Carlo simulations. The Metropolis algorithm works as follows. If $\Delta E < 0$, i.e. if the new proposed state is better, energy-wise, than the current state μ, then the move is accepted automatically. If, on the other hand, $\Delta E > 0$, than the trial move is accepted with probability $\exp(-\beta \Delta E)$. In other words, the move from μ to ν is accepted with probability

$$p = A(\mu \to \nu) = \min\{1, \exp(-\beta \Delta E)\}. \tag{11}$$

To implement Eq. 11, we need to draw a random number r from 0 to 1, and accept the move provided that $r \leq p$.

1.4 Why the Metropolis algorithm works: ergodicity and detailed balance

Let us now try and prove why the Metropolis algorithm together with the Glauber dynamics provides a suitable way to generate states according to the Boltzmann weight in the Ising model. To do this, we need to convince ourselves that each state, μ, is generated, at least after many steps in the Markov chain (after which memory of the initial condition is lost), with probability $\sim \exp(-\beta E_\mu)$. We note that a consequence of this requirement is that all states with $E_\mu < +\infty$ (so the ones which are not disallowed thermodynamically) *must* be reachable through the Markov chain we set up, as their probability in steady state is larger than 0 (even though it may be very small if E_μ is much larger than the ground state energy, i.e. the minimum value of the energy over all configurations). This requirement is known as ergodicity, and it sometimes is very hard to prove rigorously for Monte-Carlo algorithms! In our case the Glauber dynamics is ergodic as each state can theoretically be reached from any other state by choosing and flipping a suitable number of spins.

We now write down an equation for the probability of being in state μ at

time t. This is governed by the so-called master equation,

$$\frac{\partial p_\mu}{\partial t} = -\sum_{\nu \neq \mu} p_\mu P(\mu \to \nu) + \sum_{\nu \neq \mu} p_\nu P(\nu \to \mu) \tag{12}$$

In the right hand side of Eq. 12, the first term represents the probability flux out of state μ, while the second term represents the flux into μ. We note that we can include the term μ in both sums, as this is equivalent to adding and subtracting the same quantity, $p_\mu P(\mu \to \mu)$ ($P(\mu \to \mu)$ is the probability that the Markov chain does not move, i.e. that the trial move is rejected). The master equation then becomes

$$\frac{\partial p_\mu}{\partial t} = -\sum_{\nu} p_\mu P(\mu \to \nu) + \sum_{\nu} p_\nu P(\nu \to \mu) \tag{13}$$

where the sum is over all ν states. Note that in the Glauber dynamics the only $\nu \neq \mu$ states which contribute are those which differ from μ by a single spin flip, so that many of the $P(\mu \to \nu)$ and $P(\nu \to \mu)$ probabilities are zero – anyway the master equation still holds.

Now, if any equilibrium is to be reached, as is required of our Markov chain (whose associate p_μ should approach the Boltzmann distribution), we need the flux out of μ to be equal to the flux into μ at large times. Therefore the following identity has to hold,

$$\sum_{\nu} p_\mu P(\mu \to \nu) = \sum_{\nu} p_\nu P(\nu \to \mu). \tag{14}$$

By noting that $\sum_\nu P(\mu \to \nu) = 1$ (i.e. something has to happen, either μ stays where it is or goes into some other state), one may rewrite this equation as $p_\mu = \sum_\nu p_\nu P(\nu \to \mu)$. While correct, this is not too helpful for us here though, as the resulting equation is an infinite matrix equation which is difficult to work with. On the other hand, one may observe that the simplest way to solve Eq. 14 is to require that all the terms in the two series are individually equal, namely to require that

$$p_\mu P(\mu \to \nu) = p_\nu P(\nu \to \mu), \tag{15}$$

for all possible states ν and μ. Eq. 15 is the so-called detailed balance condition, and is used ubiquitously in Monte-Carlo simulations (even though as clear from this derivation it is not strictly necessary for a Markov chain to reach equilibrium).

Now, for Glauber dynamics, Eq. 15 is not trivial only for states μ and ν which differ by (at most) one spin flip. For those, the probabilities are non-zero and we can rewrite Eq. 15 as

$$\frac{P(\nu \to \mu)}{P(\mu \to \nu)} = \frac{p_\mu}{p_\nu} = \exp\left[-\beta\left(E_\mu - E_\nu\right)\right], \tag{16}$$

where the last equality stems from the fact that we want to converge to the Boltzmann distribution.

For our Markov chain, which couples Glauber dynamics and the Metropolis algorithm, the probability $P(\nu \to \mu)$ is given by:

$$P(\nu \to \mu) = g(\nu \to \mu) A(\nu \to \mu) \tag{17}$$

where $g(\nu \to \mu)$ is the probability of choosing μ as a proposed new state starting from μ, while $A(\nu \to \mu)$ is the acceptance probability in Eq. 11. Noting that $g(\mu \to \nu) = g(\nu \to \mu) = 1/N$ (where N is the number of Ising spins), as the spin to be flipped is chosen randomly with uniform probability, we obtain that

$$\begin{aligned} \frac{P(\nu \to \mu)}{P(\mu \to \nu)} &= \frac{g(\nu \to \mu) A(\nu \to \mu)}{g(\mu \to \nu) A(\mu \to \nu)} \\ &= \frac{1/N \min\{1, \exp\left[-\beta(E_\mu - E_\nu)\right]\}}{1/N \min\{1, \exp\left[-\beta(E_\nu - E_\mu)\right]\}} \\ &= \exp\left[-\beta\left(E_\mu - E_\nu\right)\right], \end{aligned} \tag{18}$$

as required. This proves that the Glauber dynamics and the Metropolis algorithm provide a sound way to set up a Monte-Carlo simulations to sample the Boltzmann distribution for an Ising model.

An alternative way to sample an equilibrium state of the Ising model is to use *Kawasaki dynamics*. Here, one chooses two distinct sites i and j (not necessarily neighbours), and considers as a trial move exchanging this pair of spins. You should also implement this choice in Checkpoint 1, and think about its effects. Is Kawasaki dynamics ergodic? What ensemble of states does it sample efficiently?

1.5 Pseudo-random number generators

Now that we have set up in principle a Monte-Carlo simulation, it is useful to briefly pause and ask, how can we generate random numbers? Luckily, it turns out that the only random number we really need to generate are uniformly distributed between 0 and 1 – any distribution of random numbers which we require can be, with a variable degree of intermediate work, be mapped onto the uniform distribution between 0 and 1!

Even so, it is far from trivial to generate random numbers efficiently. The current algorithms actually use some deterministic recursions to generate these numbers, so that these algorithms are usually called "pseudo-random number" generators. The simplest way to generate pseudo-random numbers is to define a succession of integers $\{i_n\}_n$, such that

$$i_n = f(i_{n-1}) \tag{19}$$

where f is some function. The most common pseudo-random number generators are the "linear congruential generators", which work with the following recursion

$$i_n = (a i_{n-1} + c) \bmod m \tag{20}$$

where a, c, and m are integers, and where $p \bmod q$ returns the remainder after dividing p by q (e.g. $5 \, mod \, 3 = 2$). If a, c and m are appropriately chosen, dividing

i_n by m one gets a "random number" between 0 and 1. A recommended choice with this simple algorithm (see the very good book on Monte-Carlo simulations by Barkema and Newman) has $a = 2416$, $c = 374441$ and $m = 1771875$.

Note that the fact that the pseudo-random numbers are defined via a deterministic recursion has the consequence that after a period w the sequence repeats. In practice, provided that w is large enough this is not really an issue. The Python random number generator is good enough for our purposes for this course. A very good random number generator is Mersenne-Twister, which has a really long recurrence period.

1.6 How to study phase transitions in finite systems

OK: now we have a sound method (the Metropolis algorithm) which allows us to set up a Markov chain which is guaranteed to converge, after a sufficient amount of time (the so-called "equilibration time") has elapsed, to the Boltzmann distribution. We can then estimate for instance the average (scaled) magnetisation of the system by performing n measurements (of the system in a set of states $\{\mu_i\}$) in a Monte-Carlo simulation as follows

$$\begin{aligned} m &= \frac{M}{N} = \frac{\sum_{i=1}^{N} S_i}{N} \\ \langle m \rangle &= \frac{\sum_{i=1}^{n} m(\mu_i)}{n}. \end{aligned} \tag{21}$$

We would expect that the order parameter should be close to 0 for large temperature where the system is disordered (entropy wins over J), while it should be large and close to 1 in absolute value for very small temperature (where the Boltzmann weight is dominated by the ground state). Statistical physics tells us that these regimes are separated by a phase transition at which the value of the magnetisation departs from zero, and it has a discontinuous derivative there.

Here there is, however, the first problem. In the absence of an external magnetic field there is no way to tell whether the system will select a positive or a negative magnetisation, as these are degenerate ground states. Worse still, for any finite system it is possible to go from one state to the other by flipping all spins – this will take a very long time if N is large, but it is possible, and will sooner or later occur! Therefore, the average of the magnetisation is actually zero in all cases, and it would appear impossible to speak of any phase transition. While this is strictly speaking true for finite systems, phase transitions, which only exist at $N = \infty$, luckily leave detectable signatures at finite N as well. To begin with, the issue of the two possible magnetised states coexisting can be dealt with by measuring the absolute value of the scaled magnetisation, $\langle |m| \rangle$, instead of $\langle m \rangle$. In this case, however, there is a finite values of $\langle |m| \rangle$ even at large temperatures, as we are averaging numbers which are always positive. In the limit of $N \to \infty$, the scaled magnetisation becomes smaller and smaller, but it reaches zero only at infinite N. In other words, the phase transition point has now been smoothed out.

Again, we are fortunate that we can locate the phase transition point more accurately via another route. The key observation is that fluctuations become larger and larger close to the phase transition point (the system is "undecided" between the ordered and disordered phase as their free energy is close near criticality). These fluctuations diverge at the critical point at $N \to \infty$ (this is due to the fact that fluctuations are mathematically linked to the derivative of the magnetisation and similar quantities, which diverge at the phase transition point). For finite N, there is no actual divergence, but these fluctuations are maximal and reach a peak, which is easily detectable in simulations. Examples of fluctuations which are useful to locate the critical point in the Ising model are the susceptibility (magnetisation fluctuation) and the scaled specific heat, or specific heat per spin (energy fluctuations), defined precisely respectively as

$$\chi = \frac{1}{Nk_BT}\left(\langle M^2\rangle - \langle M\rangle^2\right), \tag{22}$$

and

$$c = \frac{C}{N} = \frac{1}{Nk_BT^2}\left(\langle E^2\rangle - \langle E\rangle^2\right). \tag{23}$$

In Checkpoint 1 you should measure χ and C to quantitatively locate the critical temperature of the Ising model.

1.7 Errors, autocorrelation times, and more advanced methods of error estimates

Now you are in a position to set up a Monte-Carlo calculation to study the phase transition in the Ising model, from a disordered to an ordered state, which is observed, for instance, by decreasing the temperature T (at $J = k_B = 1$). It is also important to calculate and report the error in the estimates of the observables (such as energy, magnetisation, specific heat etc), and here we briefly discuss how to do this in a sound way.

We start from the simplest case. Suppose we want to compute the average scaled magnetisation $\langle m\rangle = \langle M\rangle/N$ of the Ising model, and to do this we generate n configurations in a Monte-Carlo simulation. Our estimate of $\langle m\rangle$ will simply be

$$\langle m\rangle = \frac{\sum_{i=1}^{n} m_i}{n}. \tag{24}$$

What about the error? The simplest option is to say that the error should be the standard deviation of the mean,

$$\sigma_m = \sqrt{\frac{\langle m^2\rangle - \langle m\rangle^2}{n-1}}. \tag{25}$$

[Note that here as throughout this notes $\langle\cdot\rangle$ denote the mean of a quantity $\cdot$.] However this is only correct under the assumption that all n measurements are statistically independent, or uncorrelated. If you take a configuration, for instance, every spin flip, the configurations will be very correlated for a long time,

and even when averaging over several steps we still essentially always consider the same configuration – this cannot lead to any decrease in the error of our estimate for $\langle m \rangle$! The way to deal with this is to compute the autocorrelation time of our Markov chain, τ, which is physically the number of steps in our sequence of measurements after which the configurations lose correlations. The autocorrelation time can be estimated via the following average

$$\chi(t) = \langle m(t')m(t+t') \rangle - \langle m \rangle^2. \tag{26}$$

If $\chi(t)$ is normalised by its value at $t = 0$, then one expects that $\chi(t)/\chi(0) \sim \exp(-t/\tau)$, i.e. $\chi(t)$ should decay to 0 after a typical time which is of the order of τ. It can then be proved that the best estimate of the error on $\langle m \rangle$, when we use n configurations, sampled every s steps in our Markov chain, is given by

$$\sigma_m \sim \sqrt{\frac{\langle m^2 \rangle - \langle m \rangle^2}{ns} 2\tau}, \tag{27}$$

where τ is the autocorrelation time, and where we have approximated $n-1 \sim n$ (for large number of measurements): the reason behind the appearance of the factor of 2 is non-trivial and we will not discuss it here!

The final point we wish to touch is the error of some more complicated quantities, such as the scaled specific heat $c = C/N$ which we are computing in Checkpoint 1. Eq. 23 shows that the estimate of this quantity requires already the first and second moment of E. The error can in principle be computed by propagation but this is very tedious. There are two simpler methods which we recommend, and that we briefly outline at the end of this Section: the bootstrap method and the jacknife method.

The bootstrap method consists in taking our n (independent, i.e. uncorrelated) states. We then pick out n of these at random (every time we pick one out of n measurements randomly, and we can pick the same measurement several times). This is called a resampling of the data. We then compute $c = C/N$ for each of the sets of measurements as we would normally do – i.e., for each resampling – and we repeat this whole procedure several time (say k). One may prove that the error on the quantity of interest, in our case c, is given

$$\sigma_c = \sqrt{\overline{c^2} - \overline{c}^2} \tag{28}$$

where the overline indicates averaging over the k resamplings.

Another possibility is to choose the "jacknife" method. This works as follows. We start from our n uncorrelated states generated by our Monte-Carlo simulations. We compute c from there, in the usual way. Then we remove the first measurement, and recompute the scaled specific heat: we call this quantity c_1. For each $i = 1, \ldots, n$, we repeat this procedure, taking out the i-th measurements from the set, and computing c from the reduced set of $n-1$ measuremets: we call each of these estimates c_i. The error on c can then be estimated by the following formula:

$$\sigma_c = \sqrt{\sum_{i=1}^{n} (c_i - c)^2}. \tag{29}$$

You should note that while for bootstrap the number of resampling is a parameter which we can specify ($k = 1000$ should be more than sufficient in our case), in the jacknife the number of resampling k by definition always equals n, the number of measurements.

You are welcome to experiment with these two methods and choose the one you prefer to compute the error bars for your Ising model data!